# MICRO-435 Quantum and Nanocomputing

**Edoardo Charbon**
**Mariagrazia Graziano**



September 17, 2025

# Quantum Computing Syllabus (Week 1-7)

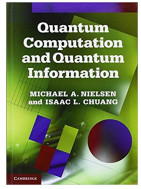


- **Fundamentals of quantum computing**
- Qubit realization & control
- Cryo-CMOS components
- Scalable quantum computers
- Quantum communication, sensing, and metrology

# Fundamentals of Quantum Computing (3)

- Basic concepts
- Qubits & superposition
- 1-qubit measurement
- 1-qubit gates
- 2-qubit systems
- 2-qubit measurement
- Bloch sphere, again
- Unitary transforms
- 2-qubit gates
- Higher-qubit gates
- Encoding functions into unitaries
- Quantum algorithms: introduction
- Quantum arithmetic
- Quantum Fourier transform, again
- Communication security: Shor's algorithm

Chapter 4



# Quantum Fourier Transform

# Conventional Discrete Fourier Transform

- Classical DFT:

  Let input vector of complex numbers $x_0, \ldots, x_{N-1}$, where $N$ is fix.
- DCT will output a vector $y_0, \ldots, y_{N-1}$, defined by

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi i j k / N}$$

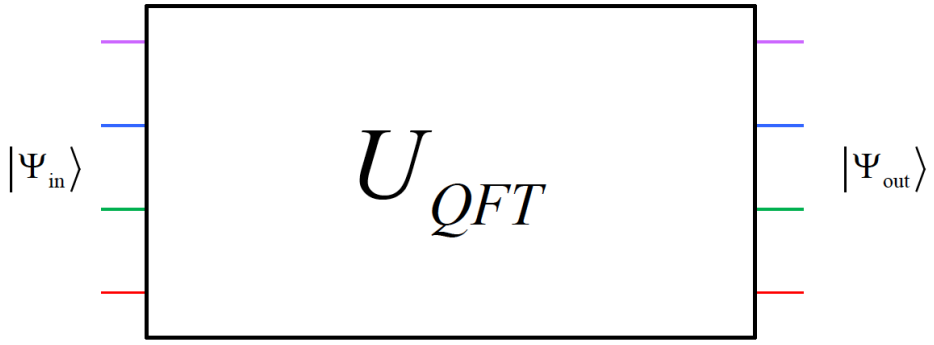
# Quantum Fourier Transform

- The quantum Fourier transform is exactly the same, except that it transforms a qubit register k into j, whereas we use the orthonormal basis $|0\rangle, \ldots, |N-1\rangle$, where $N$ is also fix.
- The QFT is defined by

$$|j\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i j k/N} |k\rangle$$

- The QFT is a unitary transformation.

# Quantum Fourier Transform




$$|\Psi_{out}\rangle = \left( \frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} \sum_{k=0}^{N-1} e^{\frac{i2\pi lk}{N}} |l\rangle\langle k| \right) |\Psi_{in}\rangle$$

$$\alpha_l' = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{\frac{i2\pi lk}{N}} \alpha_k$$

Source:
Leo DiCarlo

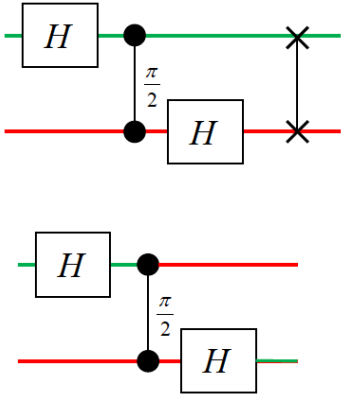
# Examples

- $N = 2$

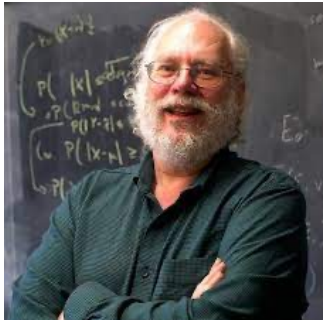
$$U_{QFT}\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix} = H$$

- $N = 4$

$$U_{QFT}\frac{1}{\sqrt{4}}\begin{pmatrix}1 & 1 & 1 & 1\\ 1 & +i & -1 & -i\\ 1 & -1 & 1 & -1\\ 1 & -i & -1 & +i\end{pmatrix}$$



# Communication Security: Shor's Algorithm



# Background: Public Key Encryption

- Problem: how can Bob send a confidential message to Alice, so as it is protected from Eve?

- Solution:
  - Alice publicly announces the encryption method and the public key $e$.
  - Alice keeps her private key $d$.
  - Bob encrypts $M$, and publicly sends the result $P$.
  - Alice decrypts $P$ using her private key $d$.

- NB:

  Only Alice can decrypt the message using her private key.

# RSA Encryption

Rivest, Shamir, and Adleman, 1977

- Alice takes 2 prime numbers $p, q$ and computes $N = pq$.
- Alice chooses $e$ coprime* with $(p-1)(q-1)$.
- Alice announces public key: $N, e$.

- Encryption: $P_i = ((M_i)^e)\ mod\ N$
- Decoding key: $d$ such that $(de) \bmod (p-1)(q-1) = 1$.
- Decryption: $M_i = \left(\left(P_i\right)^d\right) \bmod N$
- NB:
  - Checking if $e$ is coprime with $(p-1)(q-1)$ is efficient using Euclid's algorithm $O(n^3)$ .
  - Finding the modular inverse of $e$ modulo $(p-1)(q-1)$ is also efficient.
  - To crack RSA, one needs to factor $N$ into its prime factors $p$ and $q$.

*) two integers *a* and *b* are **coprime**, iff $\gcd(a, b) = 1$.
IOW the only positive integer that is a divisor of both of them is 1

# Example

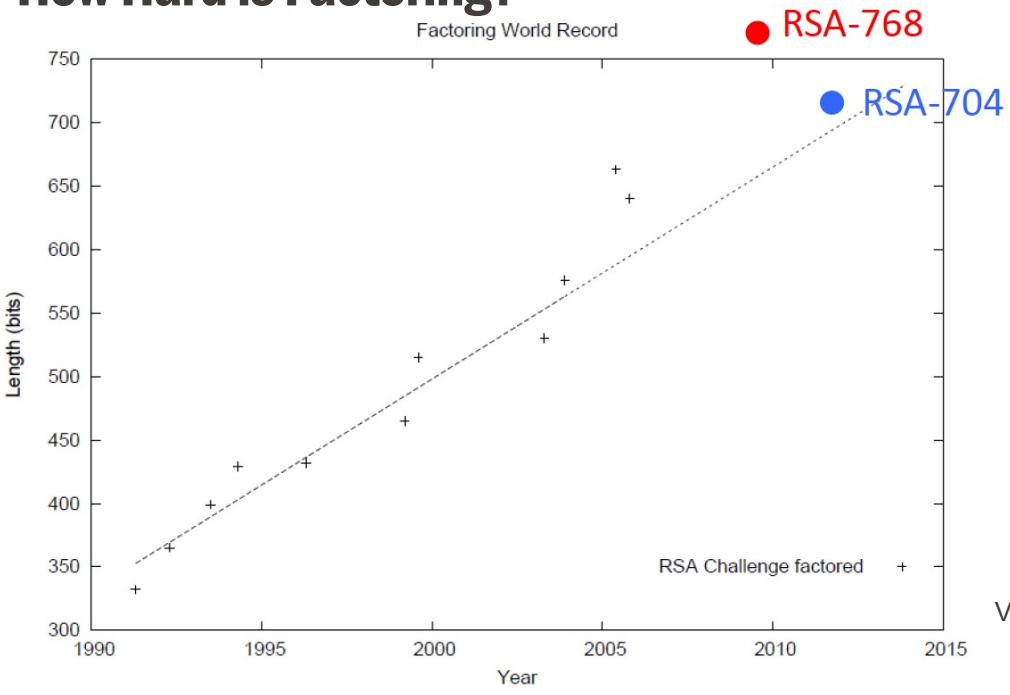
- $p = 3, q = 5$
- $N = 15$
- $(p-1)(q-1) = 8$
- Alice chooses $e = 3$, which is coprime with $8$.
- Alice announces public key: $N = 15$ and $e = 3$.

- Alice computes $d$, such as $3d \bmod 8 = 1 \longrightarrow \boldsymbol{d = 3}$.

# Example (2)

| $M$ | $\xrightarrow[e=3]{(M^e) \bmod N}$ | $P$ | $\xrightarrow[d=3]{(P^d) \bmod N}$ | $M$ |
|---|---|---|---|---|
| 0 | | 0 | | 0 |
| 1 | | 1 | | 1 |
| 2 | | 8 | | 2 |
| 3 | | 12 | | 3 |
| 4 | Notice that this | 4 | Notice that this | 4 |
| 5 | is a one-to-one | 5 | is also a one-to- | 5 |
| 6 | map | 6 | one map | 6 |
| 7 | | 13 | | 7 |
| 8 | | 2 | | 8 |
| 9 | | 9 | | 9 |
| 10 | | 10 | | 10 |
| 11 | | 11 | | 11 |
| 12 | | 3 | | 12 |
| 13 | | 7 | | 13 |
| 14 | | 14 | | 14 |

# How Hard is Factoring?




Van Meter 2006

# Number Theory Basics

- Great lecture: https://www.youtube.com/watch?v=_zTY_Rhb2Js

Umesh Vazirani (U.C. Berkeley)

Let $f(x) = (a^x) \bmod N$

- $f(x)$ is a periodic function of integer $x$, provided $a$ and $N$ are coprime.
- The period $r$, called the order of $a$ mod $N$, satisfies

$$r \leq N \quad (a^r) \bmod N = 1$$
$$(a^r - 1) \bmod N = 0$$

If $r$ is even:

$$[(a^{r/2} + 1)(a^{r/2} - 1)] \bmod N = 0$$

- Either $(a^{r/2} + 1)$ or $(a^{r/2} - 1)$ is a trivial multiple of N
- Or $\gcd(a^{r/2} + 1, N)$ and $\gcd(a^{r/2} - 1, N)$ are non-trivial factors

# Example

$N = 15$

Possible choices of $a : \{2,4,7,8,11,13,14\}$

$r \leq N \quad (a^r) \bmod N = 1$

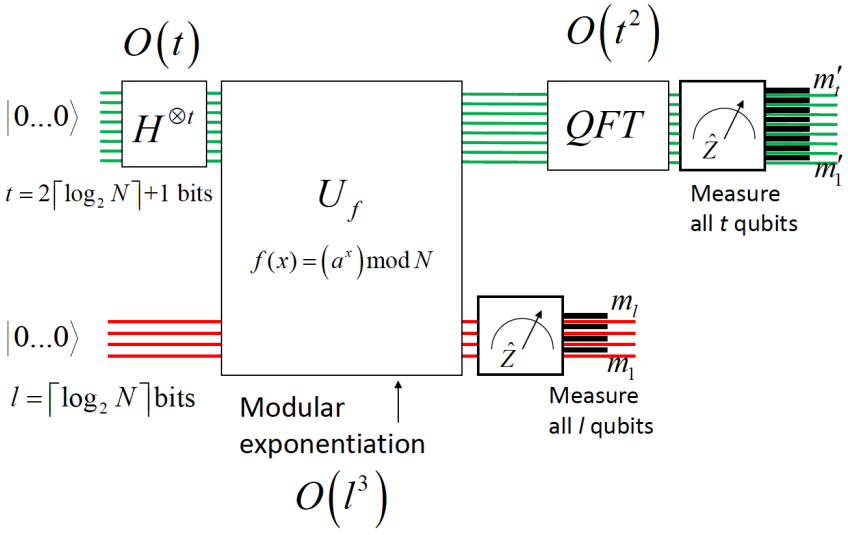
$f(x) = (a^x) \bmod 15$

| $a$ \ $x$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 1 | 2 | 4 | 8 | 1 | 2 | 4 | 8 | 1 | 2 | 4 | 8 | 1 | 2 | 4 |
| 4 | 1 | 4 | 1 | 4 | 1 | 4 | 1 | 4 | 1 | 4 | 1 | 4 | 1 | 4 | 1 |
| 7 | 1 | 7 | 4 | 13 | 1 | 7 | 4 | 13 | 1 | 7 | 4 | 13 | 1 | 7 | 4 |
| 8 | 1 | 8 | 4 | 2 | 1 | 8 | 4 | 2 | 1 | 8 | 4 | 2 | 1 | 8 | 4 |
| 11 | 1 | 11 | 1 | 11 | 1 | 11 | 1 | 11 | 1 | 11 | 1 | 11 | 1 | 11 | 1 |
| 13 | 1 | 13 | 4 | 7 | 1 | 13 | 4 | 7 | 1 | 13 | 4 | 7 | 1 | 13 | 4 |
| 14 | 1 | 14 | 1 | 14 | 1 | 14 | 1 | 14 | 1 | 14 | 1 | 14 | 1 | 14 | 1 |

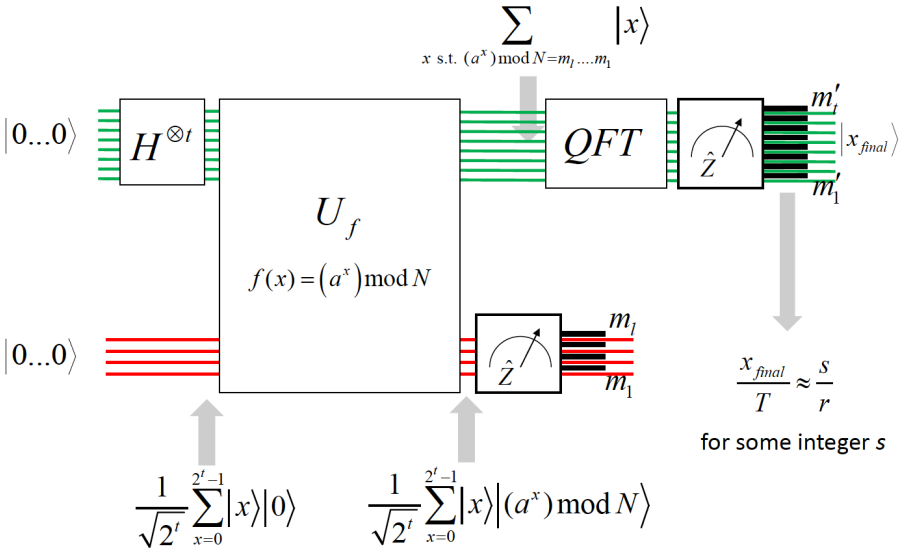
| r | $a^{r/2}-1$ | $a^{r/2}+1$ | gcd($a^{r/2}-1$,N) | gcd($a^{r/2}+1$,N) |
|---|---|---|---|---|
| 4 | 3 | 5 | 3 | 5 |
| 2 | 3 | 5 | 3 | 5 |
| 4 | 48 | 50 | 3 | 5 |
| 4 | 63 | 65 | 3 | 5 |
| 2 | 10 | 12 | 5 | 3 |
| 4 | 168 | 170 | 3 | 5 |
| 2 | 13 | 15 | 1 | 15 |

$p = 3, \; q = 5$

# Shor's Period Finding Algorithm



# Shor's Period Finding Algorithm

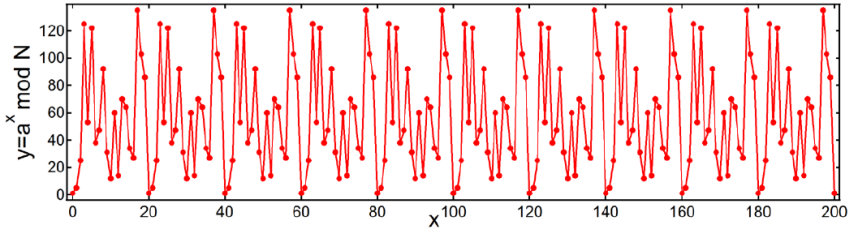


# Example: Factoring N=143

$a = 5$

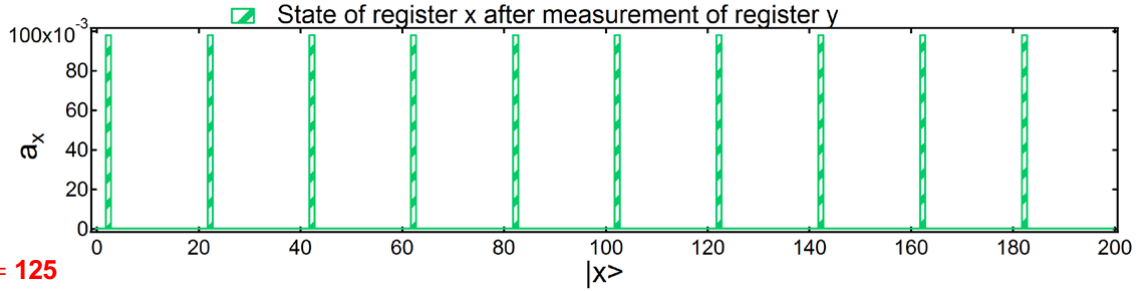
For bottom-register measurement result

$f(x) = 53$

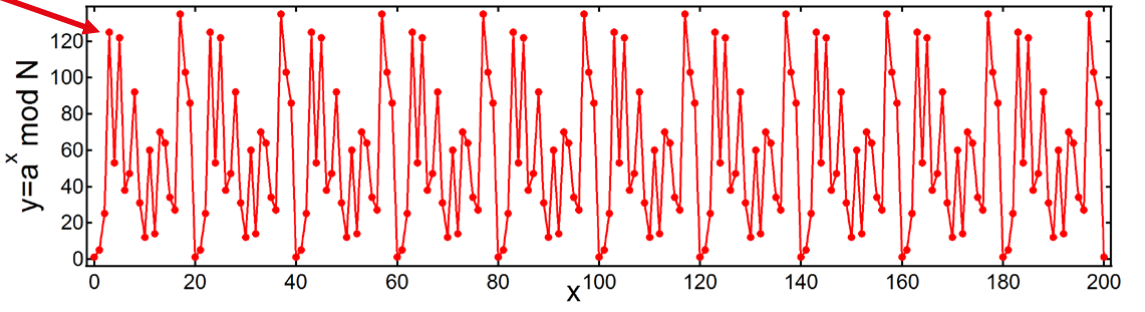



Source:
Leo DiCarlo

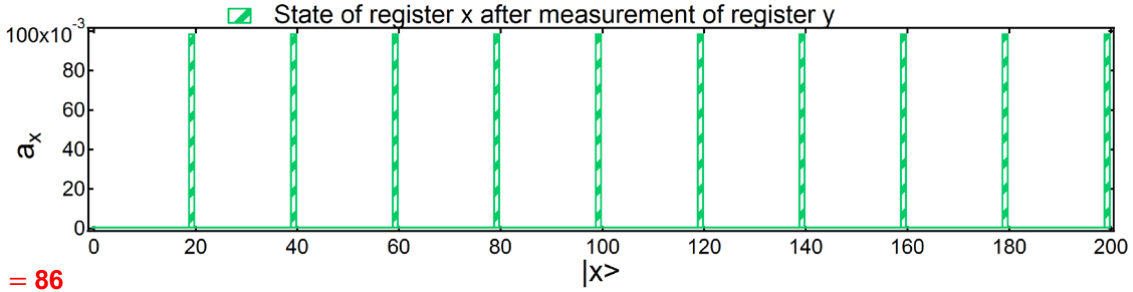
# Example: Factoring N=143



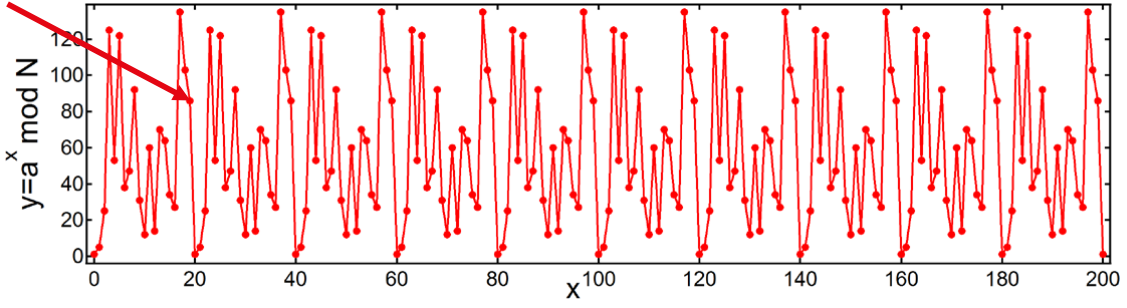

$f(x) = 125$



# Example: Factoring N=143

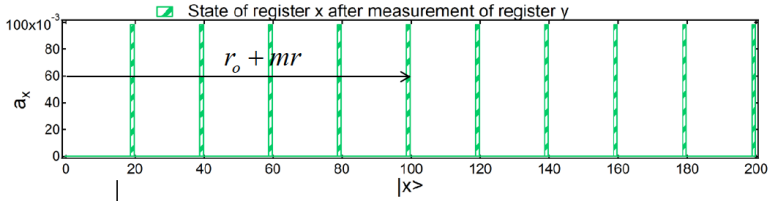



$f(x) = 86$



# Fourier Transform of Pulse Train

$a = 5$




$QFT$

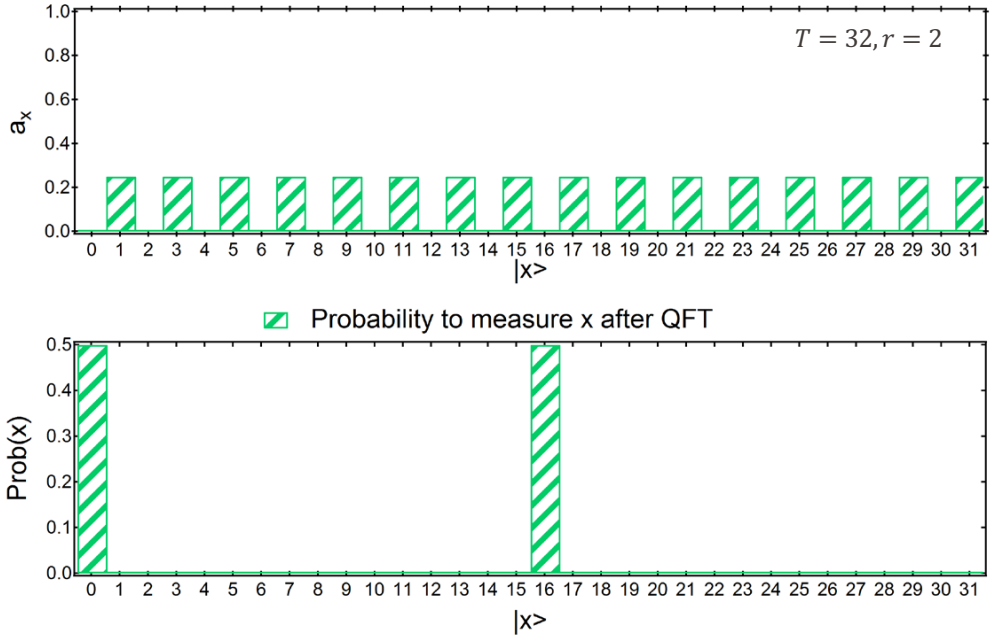
$$\alpha_x^{'} = \sum_{m=0} e^{i2\pi \frac{x(r_o+mr)}{T}} = e^{i2\pi \frac{xr_o}{T}} \sum_{m=0} e^{i2\pi \frac{xrm}{T}}$$

$$\left|\alpha_x^{'}\right|^2 = \left|\sum_{m=0} e^{i2\pi \frac{xrm}{T}}\right|^2$$

Large probability (constructive interference) for $x$ such that $xr/T$ is close to an integer.

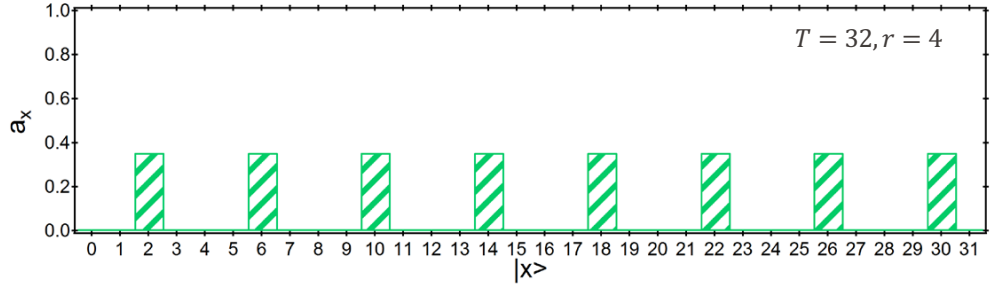
Source:
Leo DiCarlo

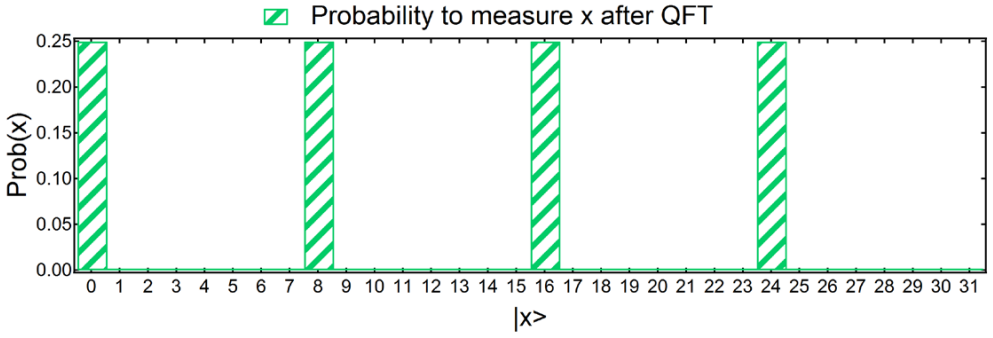
# Examples




Source:
Leo DiCarlo

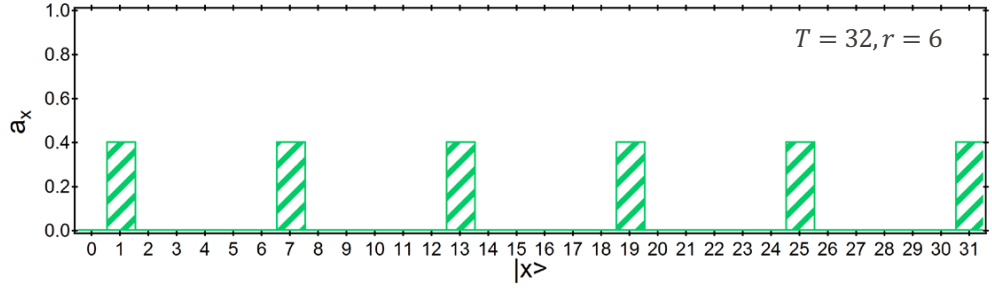
# Examples

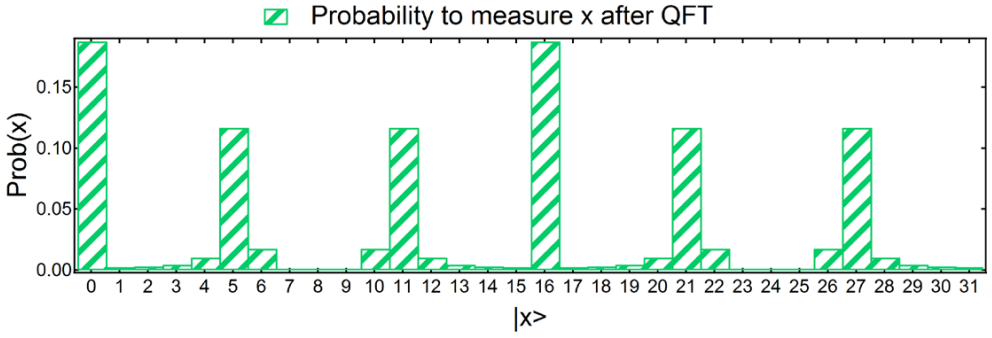



Source:
Leo DiCarlo

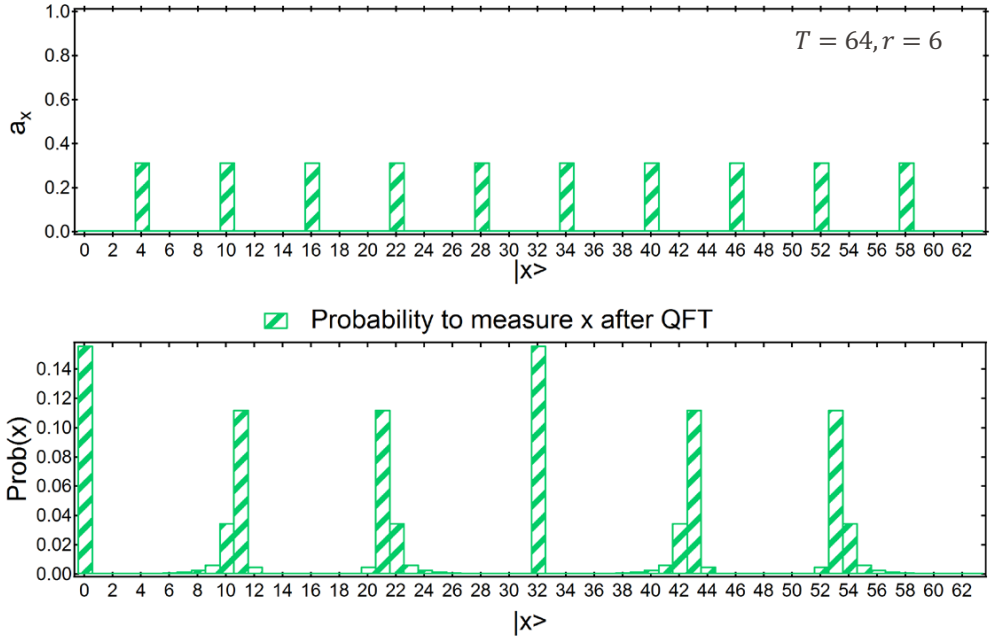
# Examples



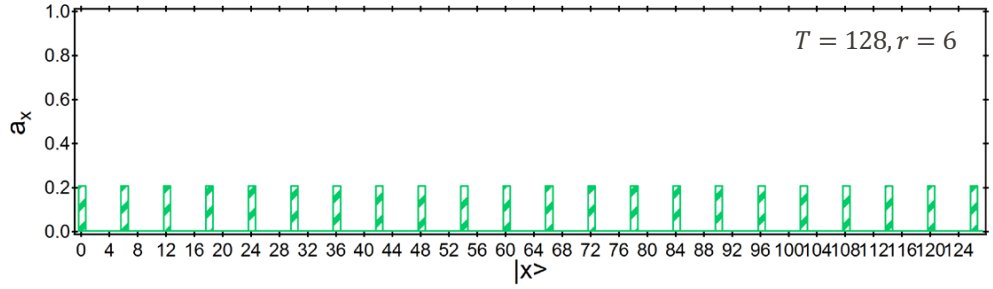

Source:
Leo DiCarlo

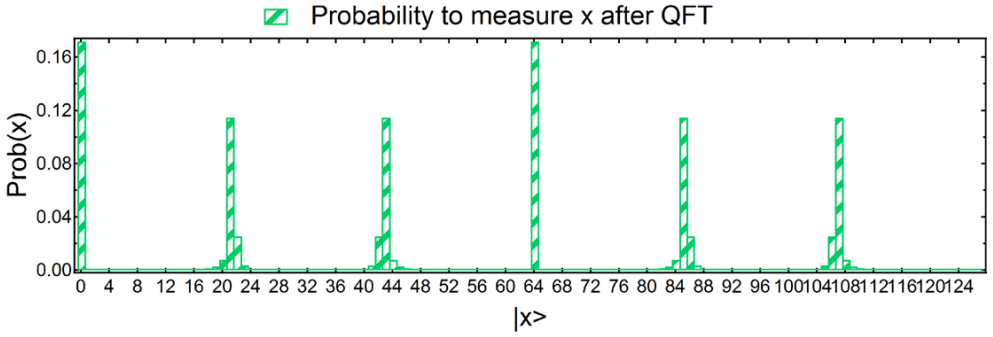
# Examples




Source:
Leo DiCarlo

# Examples

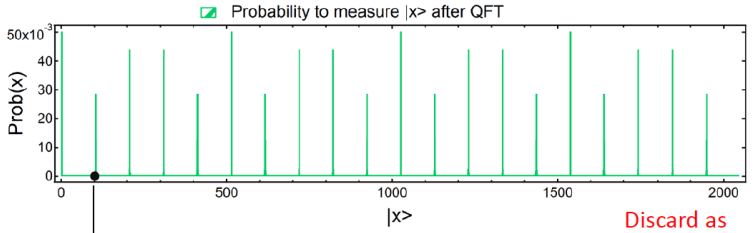



Source:
Leo DiCarlo

# Factoring N=143

$a = 5$




For top-register measurement result $x_{final}=101$

estimate of $\frac{x_{final}}{T} : \frac{101}{2048}$

Continued fractions: *1/20, 3/61, 4/81, 7/142 …*

Discard as denominator is odd

Try *r=20*.

$$a^{r/2}+1 = 9765626$$
$$a^{r/2}-1 = 9765624$$
$$\gcd(9765626, 143) = 13$$
$$\gcd(9765624, 143) = 11$$

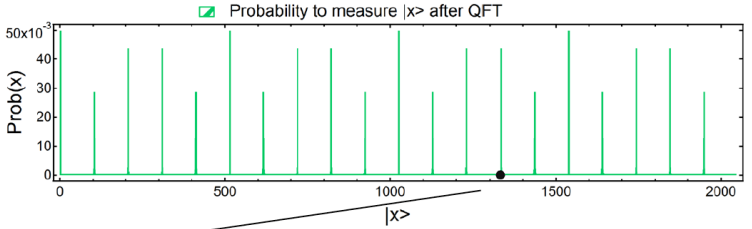
SUCCESS!

Source: Leo DiCarlo

Online continued fraction calculator:
http://www.maths.surrey.ac.uk/hosted-sites/R.Knott/Fibonacci/cfCALC.html

# Factoring N=143

$a = 5$




For top-register measurement result $x_{final}=1331$

estimate of $\frac{x_{final}}{T} : \frac{1331}{2048}$

Continued fractions: *1/2, 2/3, 11/17, 13/20…*

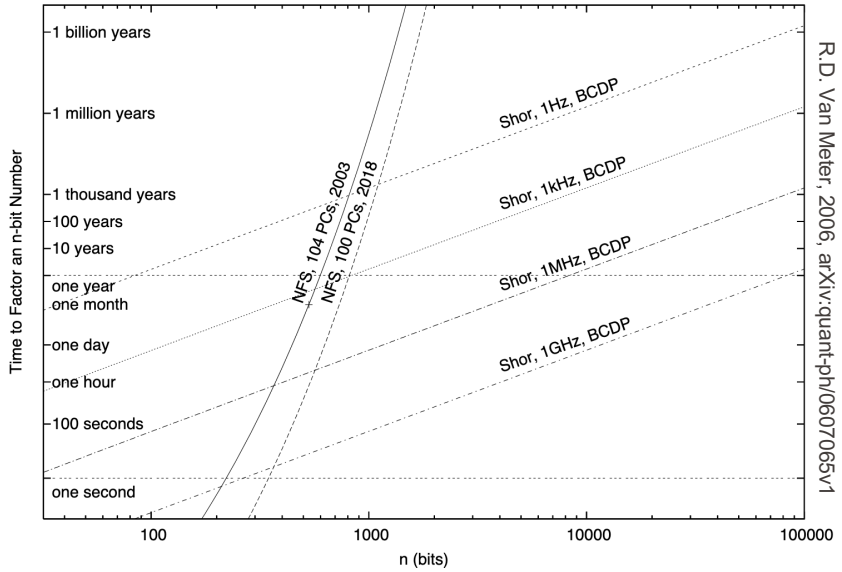
Discard as denominator is odd

Try *r=2.* $\gcd\left(a^{r/2} \pm 1, N\right) = \{1,1\}$ FAIL.

Try *r=20.* $\gcd\left(a^{r/2} \pm 1, N\right) = \{13,11\}$ SUCCESS!

Source: Leo DiCarlo

# Speedup of Shor's Algorithm




Scaling of number field sieve (NFS)

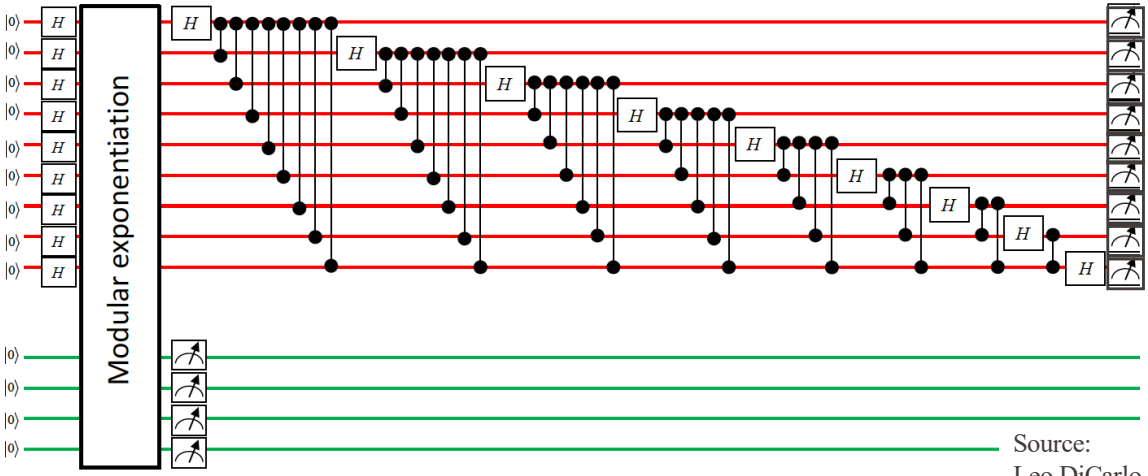
Modular exponentiation algorithm: Beckman, Chari, Devab-haktuni, and Preskill (BCDP)

# Notes

- The whole point of Shor's algorithm is to find periods of $f(x)$.
- The bottleneck of Shor's algorithm is the modular exponentiation, which requires $O(n^3)$ gates.
- The QFT for $n$ qubits is built from $O(n^2)$ gates, each of which acts on either one qubit or a pair of qubits. The QFT is efficient but the classical FFT is (relatively speaking) not.

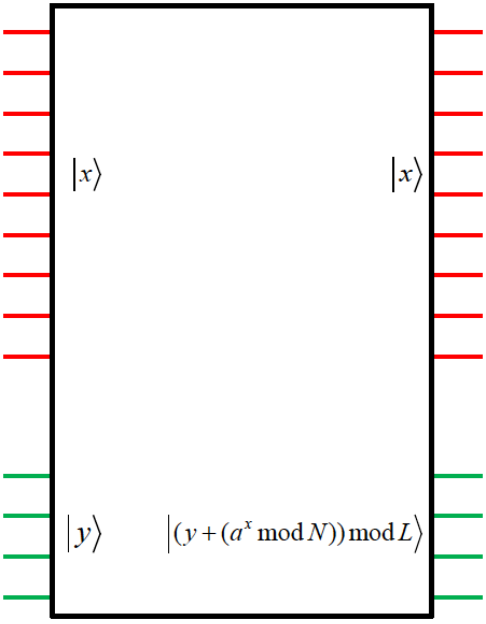
# Experimental Implementations of Shor's Algorithm

- Let $N = 15 \rightarrow t = 2\lceil \log_2 N \rceil + 1 = 9$ qubits (top register)
- $l = \lceil \log_2 N \rceil = 4$ qubits (bottom register)




Source:
Leo DiCarlo

# Quantum Unitary for Modular Exponentiation



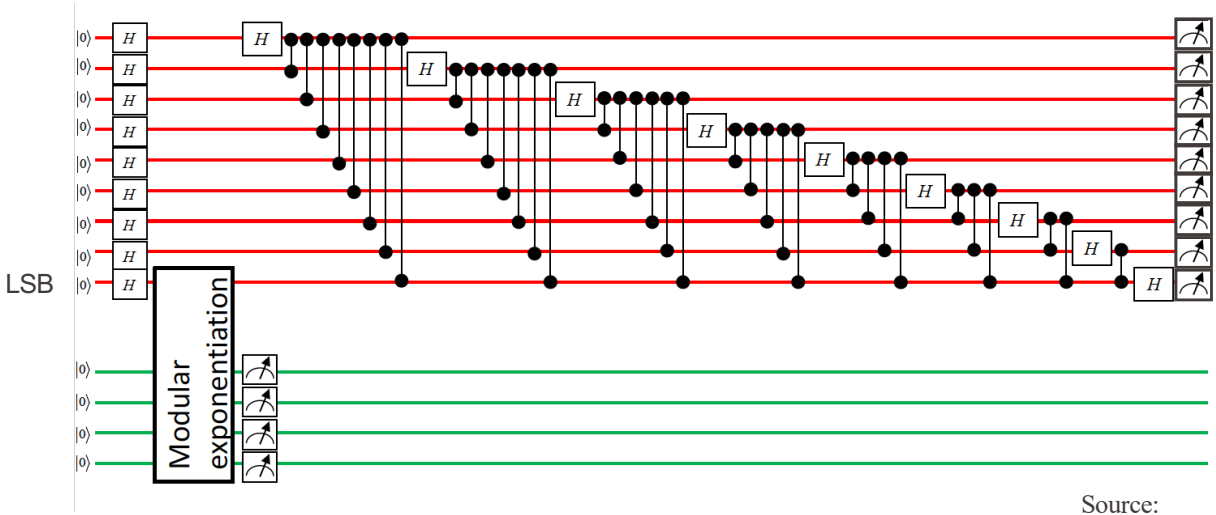
# Procedure

- Let $f(x) = a^x \bmod N$; choose $a = 4$ and $N = 15$
- $x_0$ LSB; $x_1$; $x_2$; $x_3$ MSB – All binary digits

$$x_0\colon\ a^1 = 4 \rightarrow a^1 \bmod 15 = 4 \qquad 4^{1x_0} \bmod 15 = 1 + 3x_0$$
$$x_1\colon\ a^2 = 16 \rightarrow a^2 \bmod 15 = 1 \qquad 4^{2x_1} \bmod 15 = 1$$
$$x_2\colon a^4 = 256 \rightarrow a^4 \bmod 15 = 1 \qquad 4^{4x_2} \bmod 15 = 1$$
$$x_3\colon\ a^8 = 65536 \rightarrow a^8 \bmod 15 = 1 \qquad 4^{8x_3} \bmod 15 = 1$$

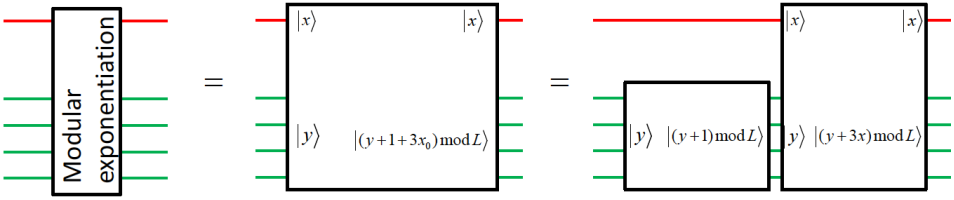
$$f(x) = 4^x \bmod 15 = 1 + 3x_0 \quad \rightarrow r = 2$$

# Compile

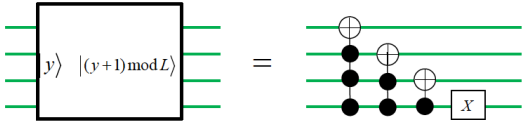


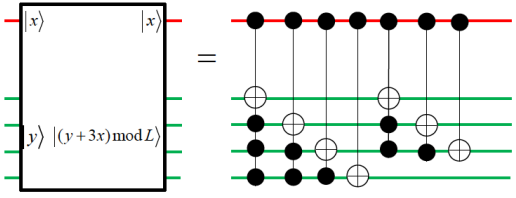

Source:
Leo DiCarlo

# Can We Reduce the Number of Oubits?



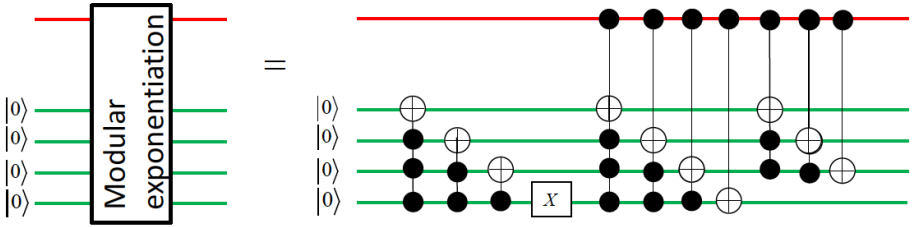

| 0 | 0 | 0 | 1 |
|---|---|---|---|
| $y_3$ | $y_2$ | $y_1$ | $y_0$ |
| $y_0 y_1 y_2 \oplus y_3$ | $y_0 y_1 \oplus y_2$ | $y_0 \oplus y_1$ | $\overline{y}_0$ |




Source:
Leo DiCarlo

# Can We Reduce the Number of Qubits Further?

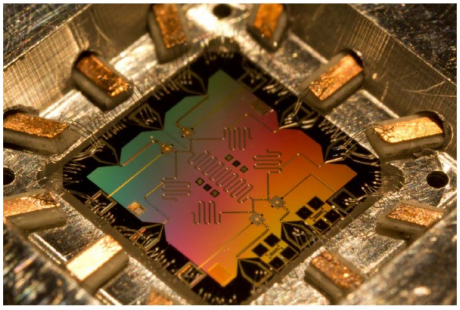


# Implementation of Shor's Algorithm in a Real QC

nature physics
LETTERS
PUBLISHED ONLINE: 19 AUGUST 2012 | DOI: 10.1038/NPHYS2385

## Computing prime factors with a Josephson phase qubit quantum processor

Erik Lucero, R. Barends, Y. Chen, J. Kelly, M. Mariantoni, A. Megrant, P. O'Malley, D. Sank, A. Vainsencher, J. Wenner, T. White, Y. Yin, A. N. Cleland and John M. Martinis*



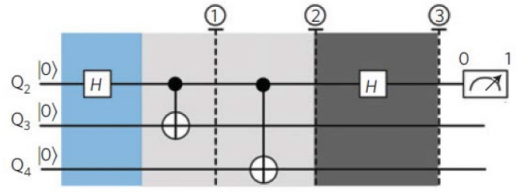
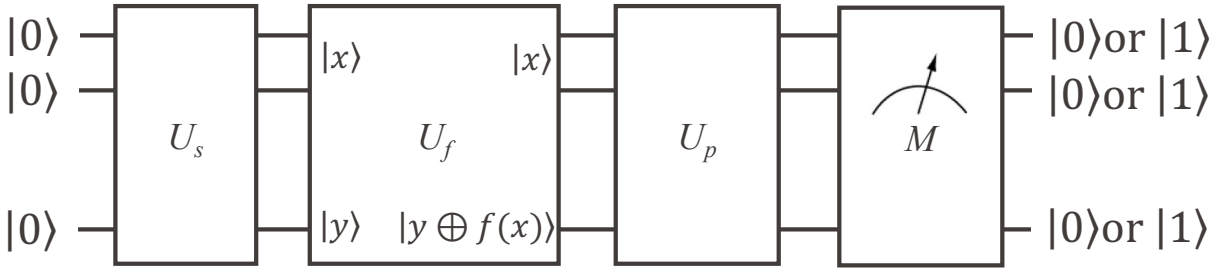



**Figure 3 | Compiled version of Shor's algorithm.**

# Review: DiVincenzo Criteria for Quantum Computing

1. A scalable physical system with well characterized qubits.
2. The ability to initialize the state of the qubits to a simple fiducial state.
3. Long relevant decoherence times.
4. A “universal” set of quantum gates.
5. A qubit-specific measurement capability.
6. The ability to interconvert stationary and flying qubits.
7. The ability to faithfully transmit flying qubits between specified locations.

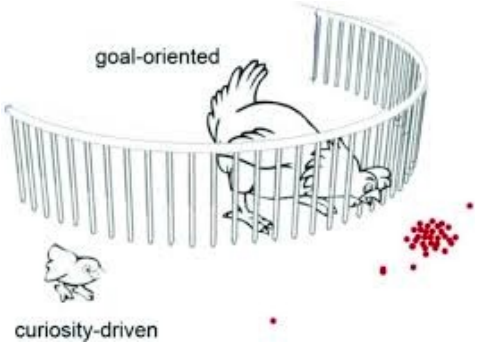
# Review: Essence of a Quantum Algorithm




**Maintain quantum coherence**

- Initialize qubits
- Create superposition
- Encode function in unitary
- Process
- Measure

# Thank you




T. Haensch